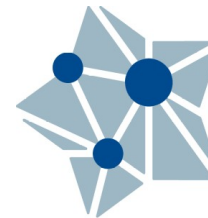


# Efficient remote interactive pipelines using CASA and Jupyter

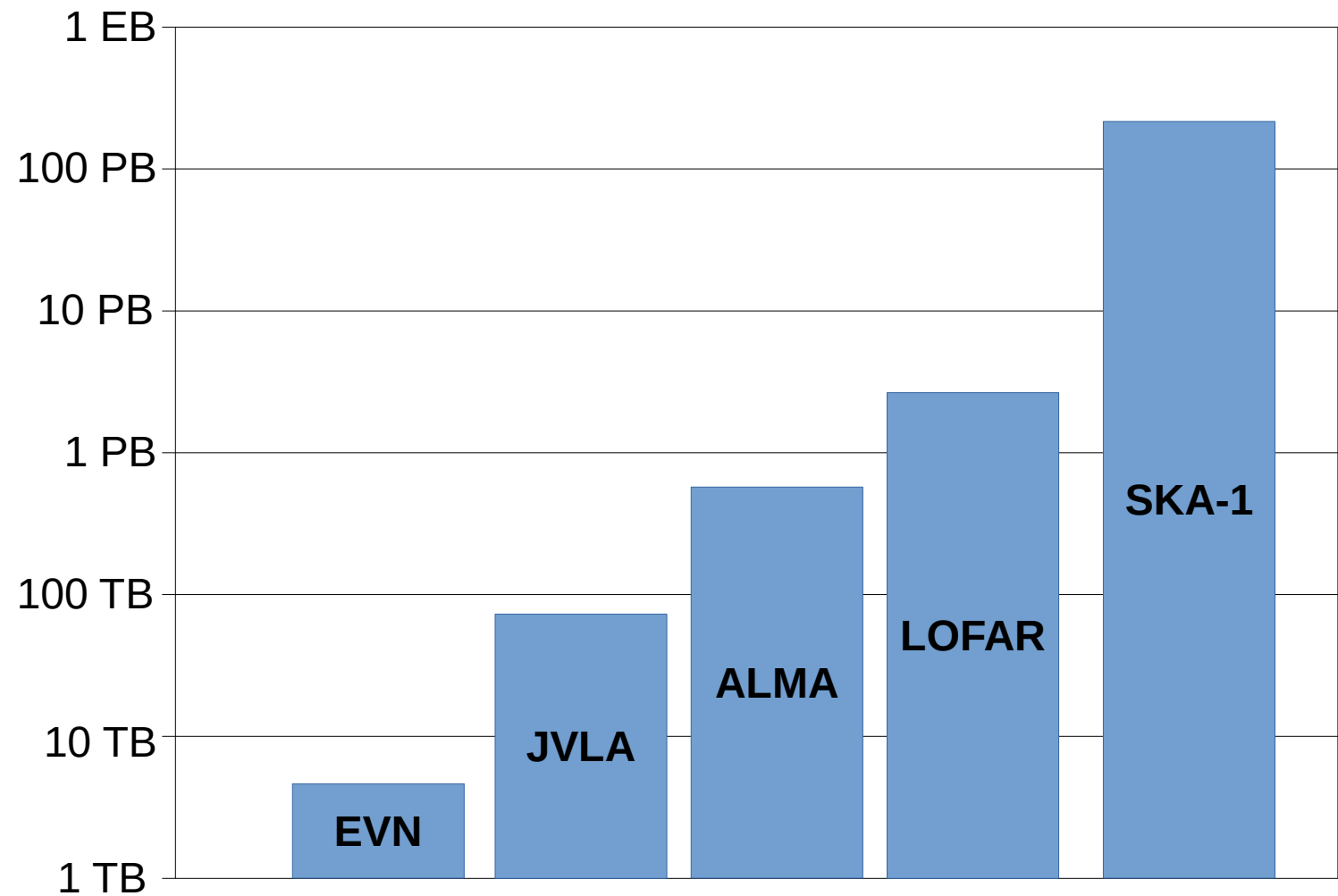
Aard Keimpema ([keimpema@jive.eu](mailto:keimpema@jive.eu))



**JIVE**  
Joint Institute for VLBI  
ERIC

H2020-Astronomy ESFRI and Research Infrastructure Cluster (Grant Agreement number: 653477).

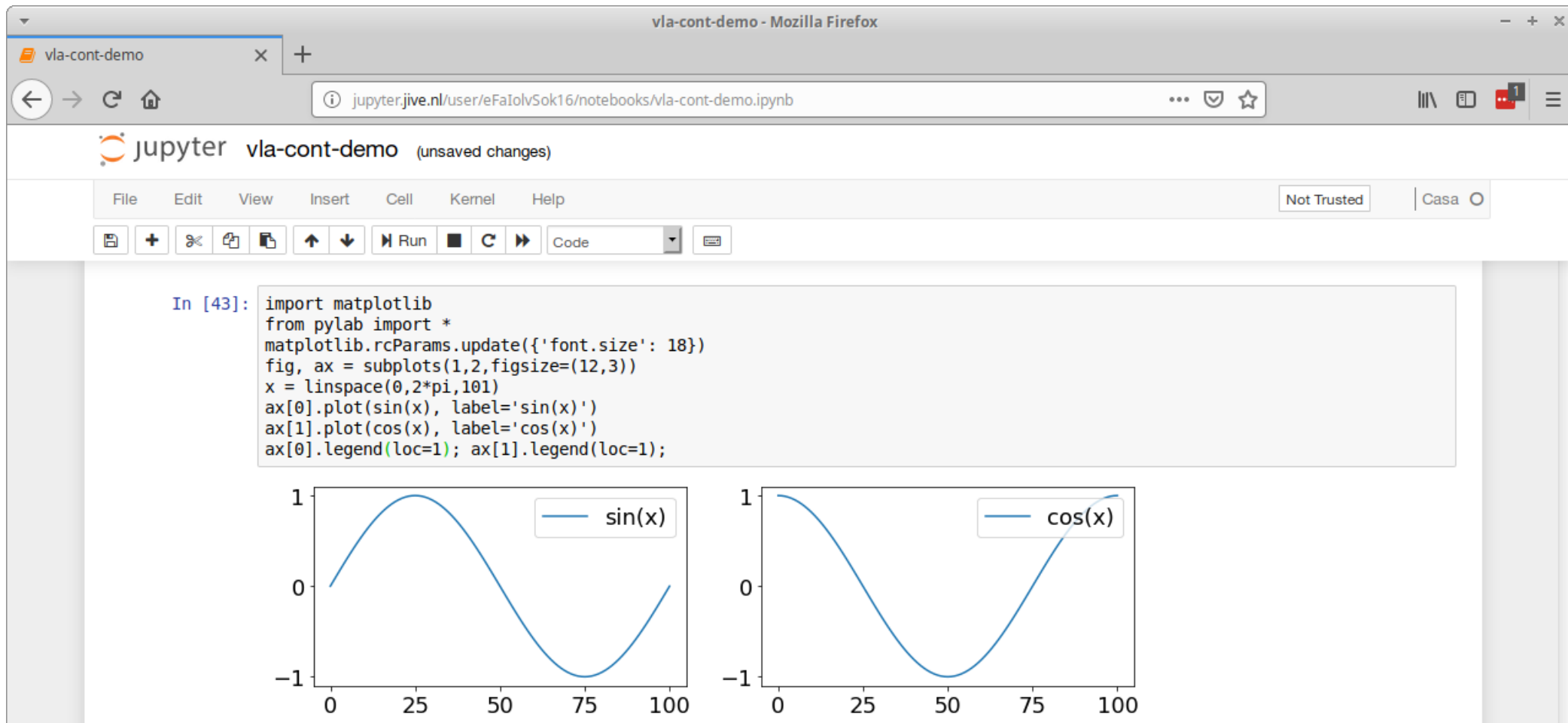
# Yearly archivable data



# Remote pipelines

- Data reduction is done where the data is stored
- CASA is the defacto standard data reduction package for radio astronomy
- Existing CASA pipelines: ALMA / JVLA, LOFAR, MEERKATHI
- Remote interactive pipelines based on Jupyter notebooks
- Successor to IPython, **CASA is based on IPython**
- Not limited to python, bindings to 40+ languages exist

# Matplotlib integration



Matplotlib figures are inlined automatically

# Annotation

**Calibration**

It is now time to begin calibrating the data. The general data reduction strategy is to derive a series of scaling factors or corrections from the calibrators, which are then collectively applied to the science data. For more discussion of the philosophy, strategy, and implementation of calibration of synthesis data within CASA, see [Synthesis Calibration](#) in the CASA Reference Manual.

Recall that the observed visibility  $V'$  between two antennas  $(i, j)$  is related to the true visibility  $V$  by:

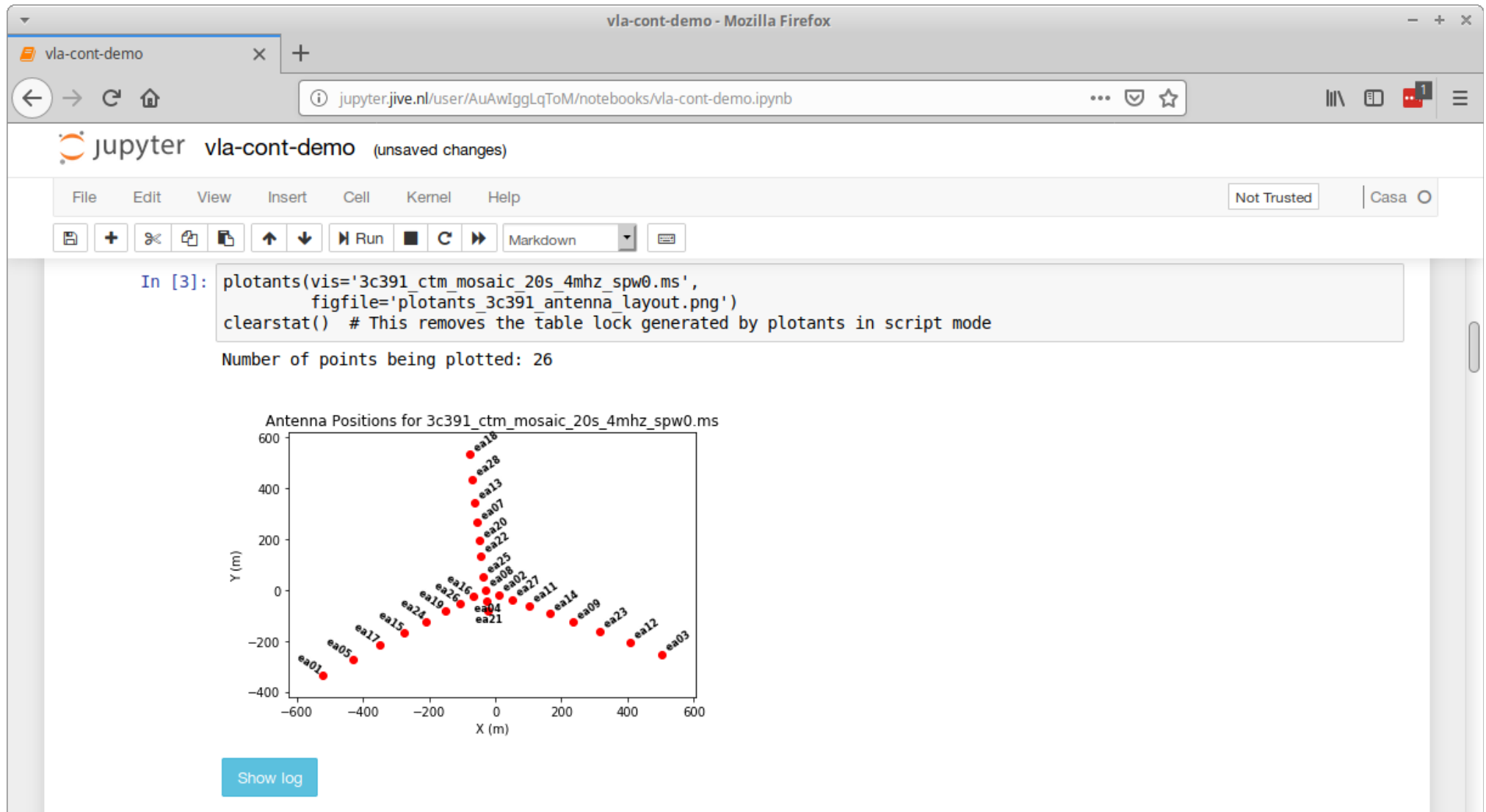
$$V'_{i,j}(u, v, f) = b_{ij}(t) [B_i(f, t) B_j^*(f, t)] g_i(t) g_j(t) V_{i,j}(u, v, f) e^{i[\theta_i(t) - \theta_j(t)]}$$

Here, for generality, we show the visibility as a function of frequency  $f$  and spatial wave numbers  $u$  and  $v$ . The other terms are:

- $g_i$  and  $\theta_i$  are the amplitude and phase portions of what is commonly termed the complex gain. They are shown separately here because they are usually determined separately. For completeness, these are shown as a function of time  $t$  to indicate that they can change with temperature, atmospheric conditions, etc.
- $B_i$  is the complex bandpass, the instrumental response as a function of frequency  $f$ . As shown here, the bandpass may also vary as a function of time.
- $b(t)$  is the often-neglected baseline term. It can be important to include for the highest dynamic range images or shortly after a configuration change at the VLA, when antenna positions may not be known well.

Notebooks can be annotated with rich text

# Integrated help



Integrated help for all functions

# Integrated help

vla-cont-demo - Mozilla Firefox

vla-cont-demo x +

jupyter.vive.nl/user/AuAwIggLqToM/notebooks/vla-cont-demo.ipynb

jupyter vla-cont-demo (autosaved)

File Edit View Insert Cell Kernel Help Not Trusted Casa

In [3]: `plotants(vis='3c391_ctm_mosaic_20s_4mhz_spw0.ms',  
figfile='plotants_3c391_antenna_layout.png')`

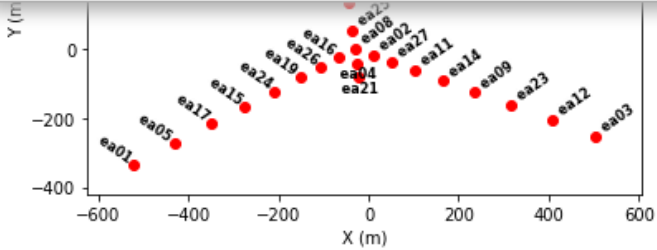
Detailed Description:

The location of the antennas in the MS will be plotted with  
X-toward local east; Y-toward local north.

Arguments :

vis: Name of input visibility file (MS)  
Default Value:

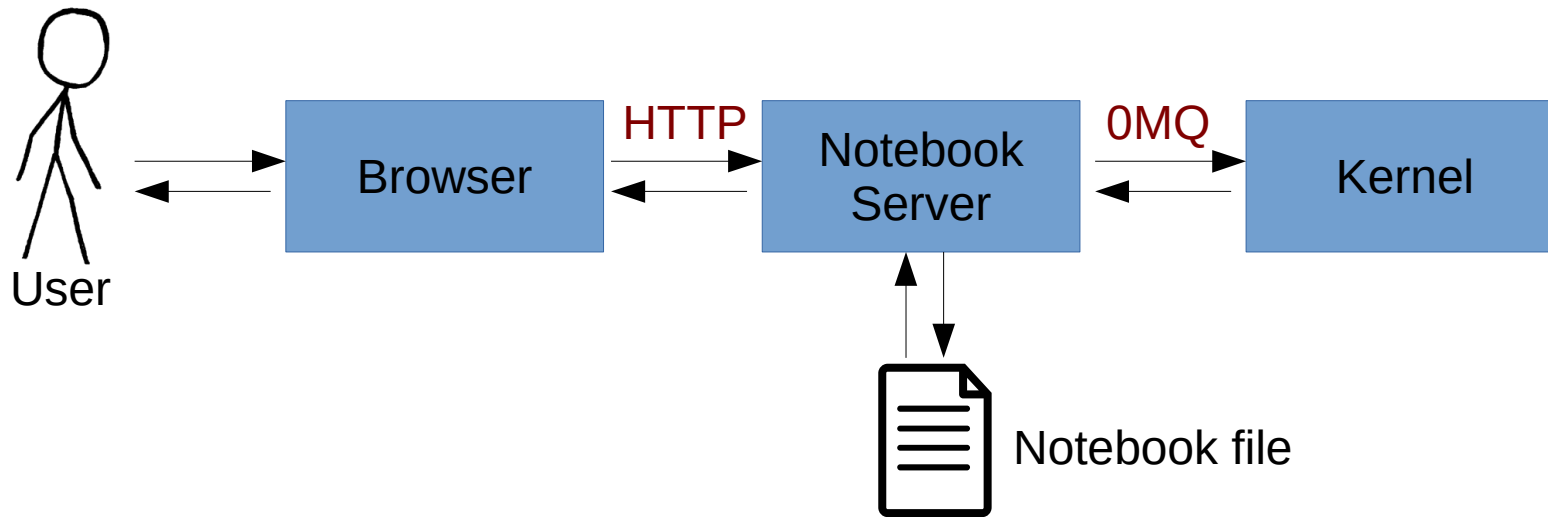
figfile: Save the plotted figure to this file



Show log

Integrated help for all functions

# Jupyter Architecture



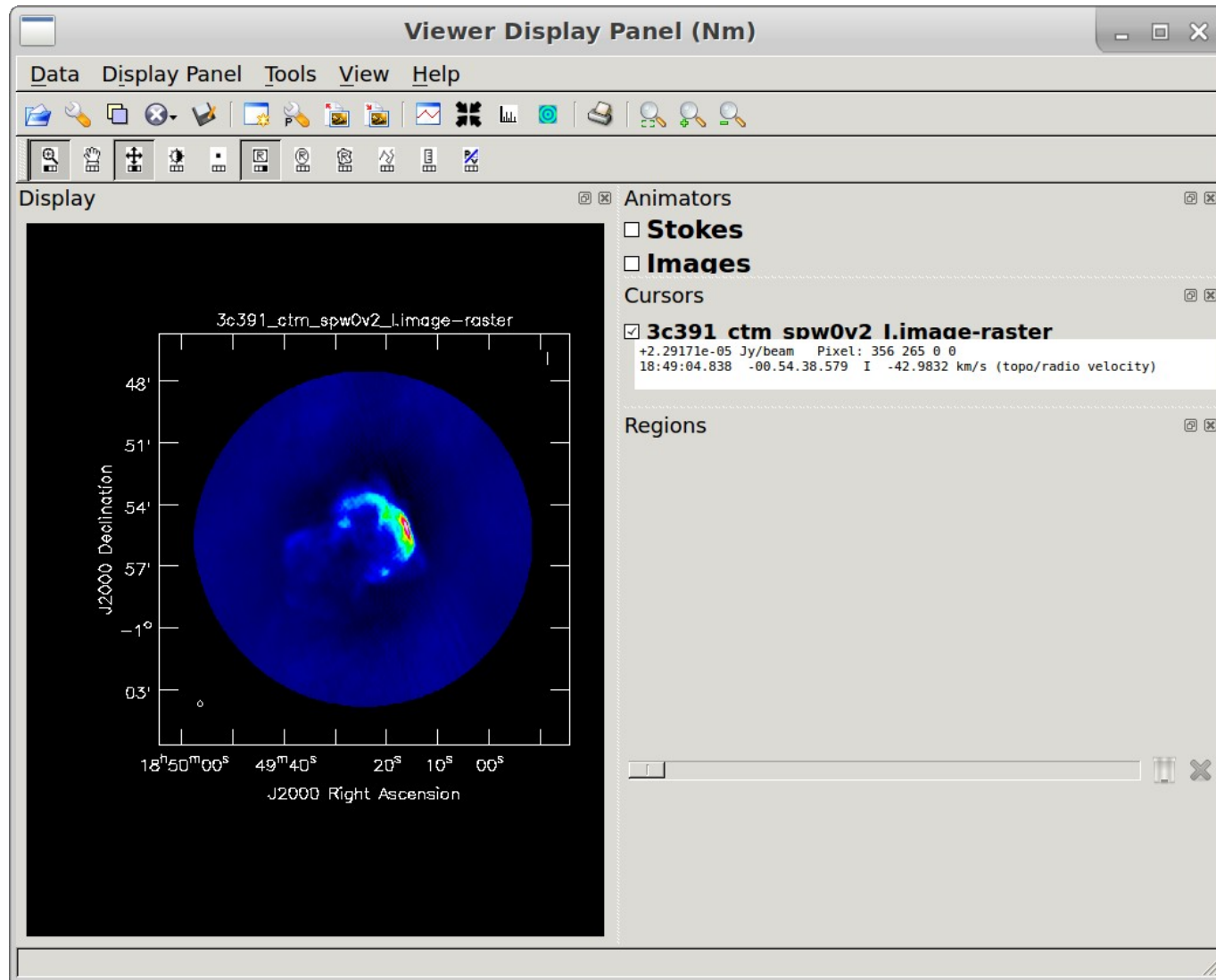
- Notebook server knows nothing about target language
- All language specifics are in the **kernel**
  - *Wrapper-kernel*: written in python, easiest to implement
  - *Native-kernel*: written in target language, much more work



# CASA Jupyter kernel

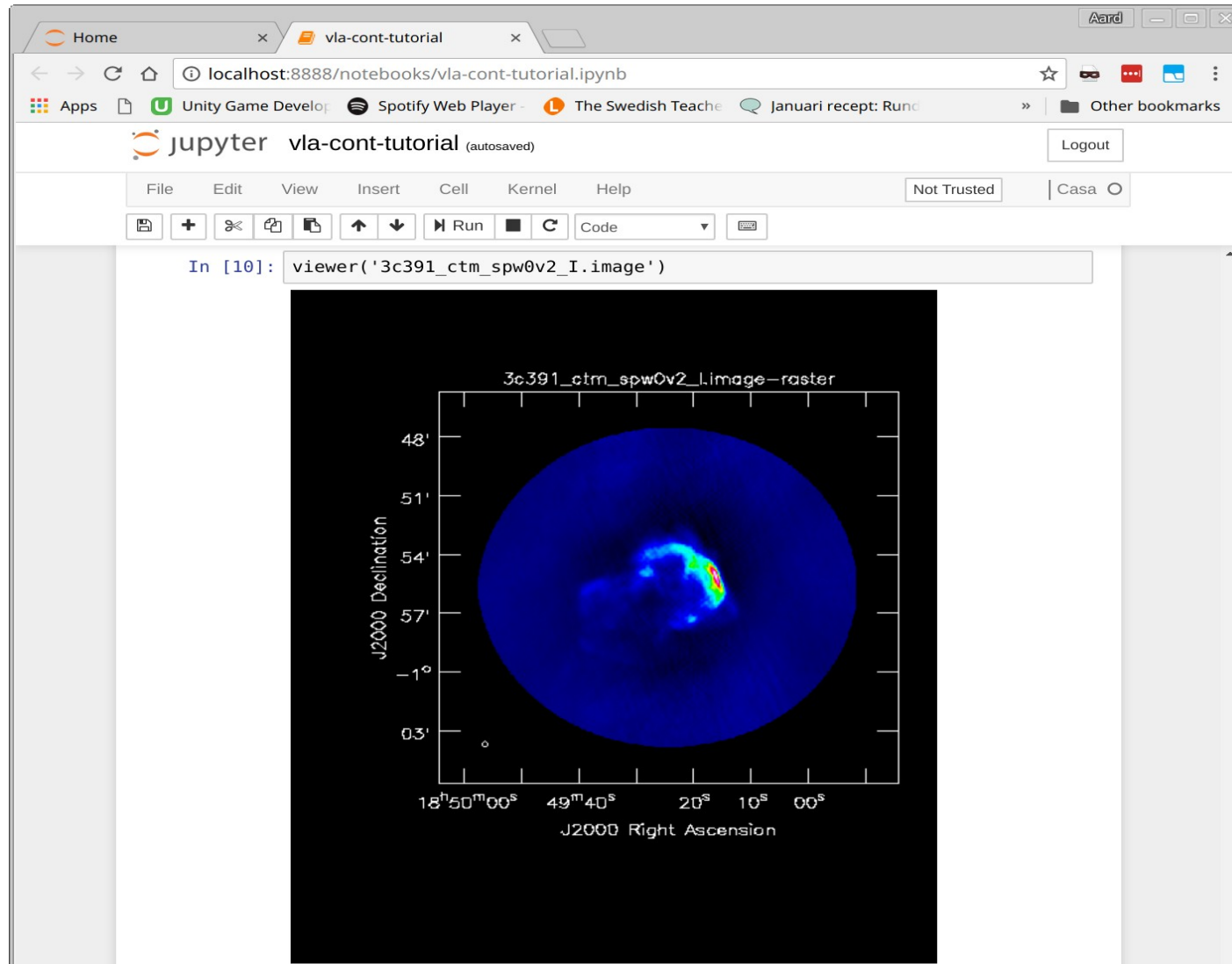
- Based on the generic python wrapper kernel
- Initialization:
  - Load needed python packages: casacore, casa tasks, matplotlib, ....
  - Setup environment: Config, logging, dbus, etc..
- CASA has python bindings for all tasks
- Many tasks open a C++ coded GUI, these are wrapped so that output goes to notebook.
- *<https://github.com/aardk/jupyter-casa>*

# Example: casaviewer



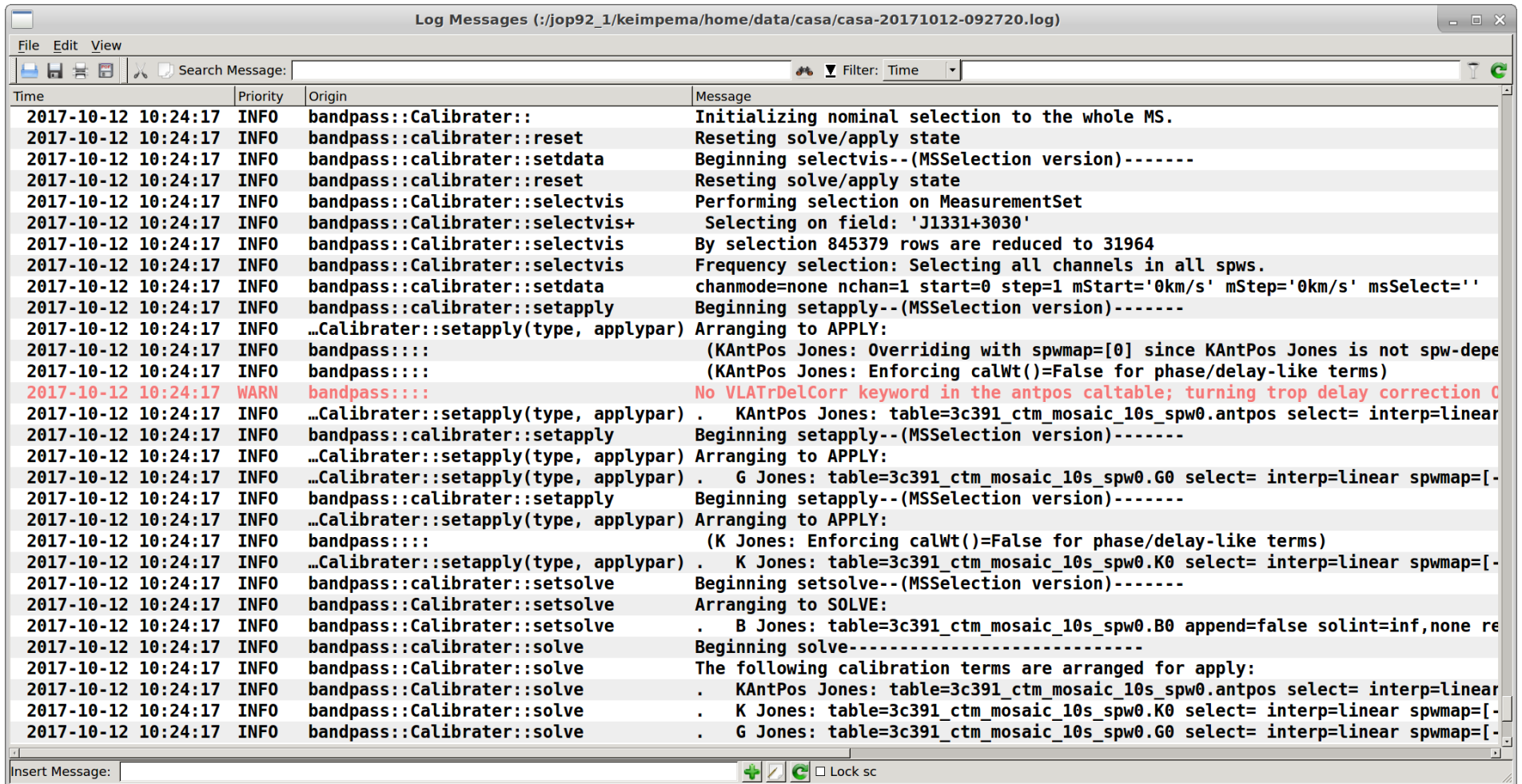
```
viewer('3c391_ctm_spw0v2_limage')
```

# Example: casaviewer



`viewer('3c391_ctm_spw0v2_I.image', gui = False,  
outformat = 'png', outfile = viewer_temp.png)`

# Logging



Time	Priority	Origin	Message
2017-10-12 10:24:17	INFO	bandpass::Calibrator::	Initializing nominal selection to the whole MS.
2017-10-12 10:24:17	INFO	bandpass::calibrator::reset	Resetting solve/apply state
2017-10-12 10:24:17	INFO	bandpass::calibrator::setdata	Beginning selectvis--(MSSelection version)-----
2017-10-12 10:24:17	INFO	bandpass::calibrator::reset	Resetting solve/apply state
2017-10-12 10:24:17	INFO	bandpass::Calibrator::selectvis	Performing selection on MeasurementSet
2017-10-12 10:24:17	INFO	bandpass::Calibrator::selectvis+	Selecting on field: 'J1331+3030'
2017-10-12 10:24:17	INFO	bandpass::Calibrator::selectvis	By selection 845379 rows are reduced to 31964
2017-10-12 10:24:17	INFO	bandpass::Calibrator::selectvis	Frequency selection: Selecting all channels in all spws.
2017-10-12 10:24:17	INFO	bandpass::calibrator::setdata	chanmode=none nchan=1 start=0 step=1 mStart='0km/s' mStep='0km/s' msSelect=''
2017-10-12 10:24:17	INFO	bandpass::calibrator::setapply	Beginning setapply--(MSSelection version)-----
2017-10-12 10:24:17	INFO	...Calibrator::setapply(type, applypar)	Arranging to APPLY:
2017-10-12 10:24:17	INFO	bandpass:::	(KAntPos Jones: Overriding with spwmap=[0] since KAntPos Jones is not spw-depe
2017-10-12 10:24:17	INFO	bandpass:::	(KAntPos Jones: Enforcing calWt()=False for phase/delay-like terms)
2017-10-12 10:24:17	WARN	bandpass:::	No VLATrDelCorr keyword in the antpos caltable; turning trop delay correction C
2017-10-12 10:24:17	INFO	...Calibrator::setapply(type, applypar)	. KAntPos Jones: table=3c391_ctm_mosaic_10s_spw0.antpos select= interp=linear
2017-10-12 10:24:17	INFO	bandpass::calibrator::setapply	Beginning setapply--(MSSelection version)-----
2017-10-12 10:24:17	INFO	...Calibrator::setapply(type, applypar)	Arranging to APPLY:
2017-10-12 10:24:17	INFO	...Calibrator::setapply(type, applypar)	. G Jones: table=3c391_ctm_mosaic_10s_spw0.G0 select= interp=linear spwmap=[-
2017-10-12 10:24:17	INFO	bandpass::calibrator::setapply	Beginning setapply--(MSSelection version)-----
2017-10-12 10:24:17	INFO	...Calibrator::setapply(type, applypar)	Arranging to APPLY:
2017-10-12 10:24:17	INFO	bandpass:::	(K Jones: Enforcing calWt()=False for phase/delay-like terms)
2017-10-12 10:24:17	INFO	...Calibrator::setapply(type, applypar)	. K Jones: table=3c391_ctm_mosaic_10s_spw0.K0 select= interp=linear spwmap=[-
2017-10-12 10:24:17	INFO	bandpass::calibrator::setsolve	Beginning setsolve--(MSSelection version)-----
2017-10-12 10:24:17	INFO	bandpass::Calibrator::setsolve	Arranging to SOLVE:
2017-10-12 10:24:17	INFO	bandpass::Calibrator::setsolve	. B Jones: table=3c391_ctm_mosaic_10s_spw0.B0 append=false solint=inf,none re
2017-10-12 10:24:17	INFO	bandpass::calibrator::solve	Beginning solve-----
2017-10-12 10:24:17	INFO	bandpass::Calibrator::solve	The following calibration terms are arranged for apply:
2017-10-12 10:24:17	INFO	bandpass::Calibrator::solve	. KAntPos Jones: table=3c391_ctm_mosaic_10s_spw0.antpos select= interp=linear
2017-10-12 10:24:17	INFO	bandpass::Calibrator::solve	. K Jones: table=3c391_ctm_mosaic_10s_spw0.K0 select= interp=linear spwmap=[-
2017-10-12 10:24:17	INFO	bandpass::Calibrator::solve	. G Jones: table=3c391_ctm_mosaic_10s_spw0.G0 select= interp=linear spwmap=[-

CASA displays logging information inside *casalogger* task.

# Logging

PythonDataScienceHar x vla-cont-tutorial x

localhost:8888/notebooks/vla-cont-tutorial.ipynb

Apps Unity Game Develop Spotify Web Player The Swedish Teache Januari recept: Rund South Indian-Style V Bookmarks Other bookmarks

jupyter vla-cont-tutorial Last Checkpoint: 38 minutes ago (autosaved) Logout

File Edit View Insert Cell Kernel Help Not Trusted Casa

In [27]:

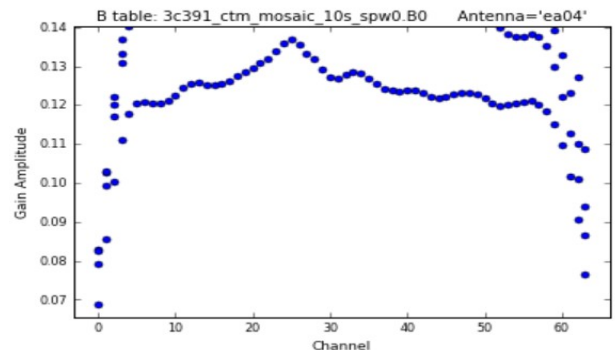
```
bandpass(vis='3c391_ctm_mosaic_10s_spw0.ms', caltable='3c391_ctm_mosaic_10s_spw0.B0',
         field='J1331+3030', spw='', refant='ea21', combine='scan',
         solint='inf', bandtype='B',
         gaintable=['3c391_ctm_mosaic_10s_spw0.antpos',
                   '3c391_ctm_mosaic_10s_spw0.G0',
                   '3c391_ctm_mosaic_10s_spw0.K0'])
```

Show log

In [28]:

```
plotcal(caltable='3c391_ctm_mosaic_10s_spw0.B0', poln='R',
        xaxis='chan', yaxis='amp', field='J1331+3030', subplot=221,
        iteration='antenna', figfile='plotcal_3c391-3C286-B0-R-amp.png')
```

B table: 3c391\_ctm\_mosaic\_10s\_spw0.B0 Antenna='ea04'

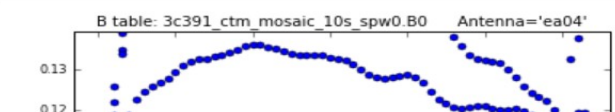


Show log

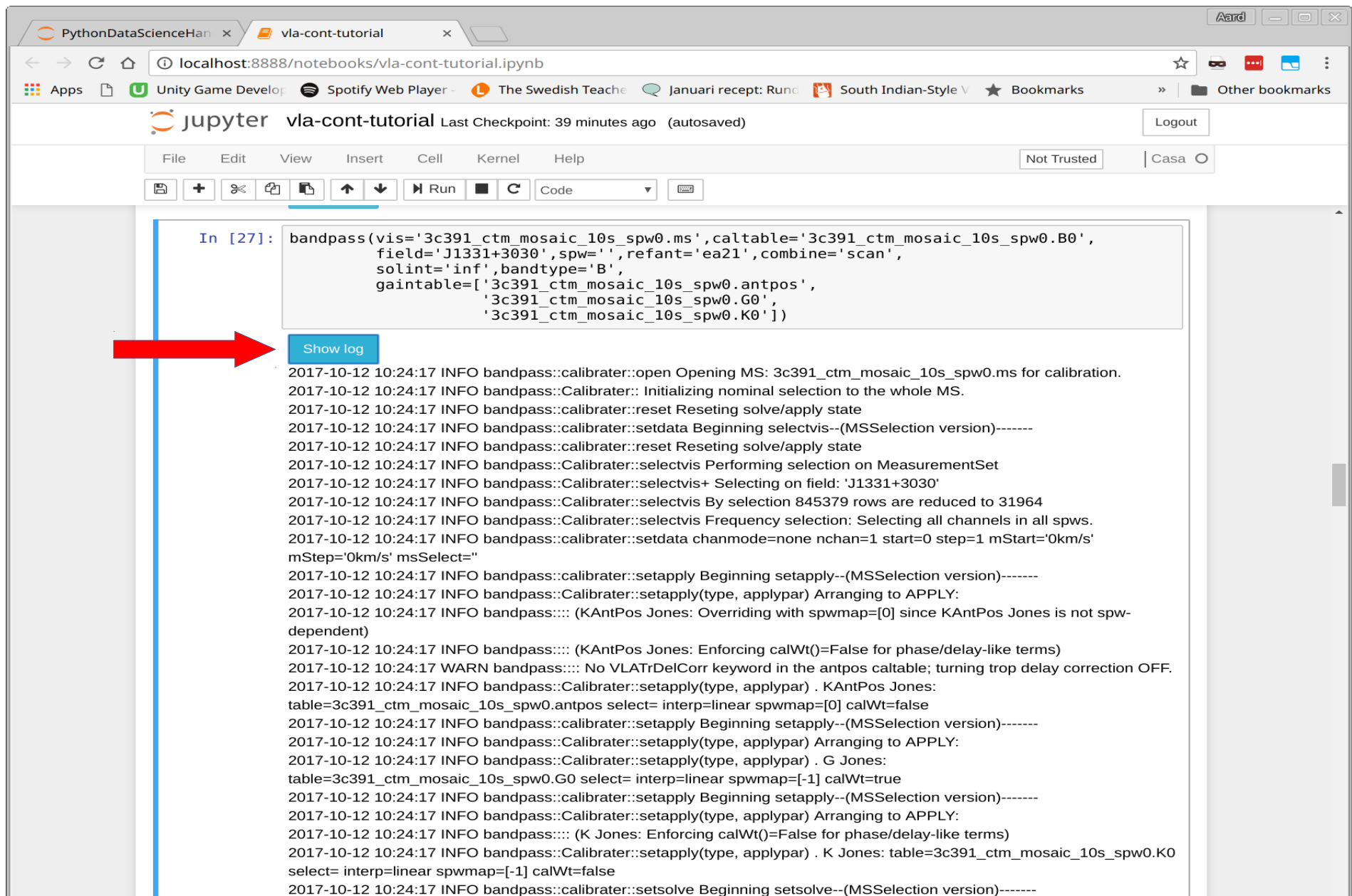
In [29]:

```
plotcal(caltable='3c391_ctm_mosaic_10s_spw0.B0', poln='L',
        xaxis='chan', yaxis='amp', field='J1331+3030', subplot=221,
        iteration='antenna', figfile='plotcal_3c391-3C286-B0-L-amp.png')
```

B table: 3c391\_ctm\_mosaic\_10s\_spw0.B0 Antenna='ea04'



# Logging



The screenshot shows a Jupyter Notebook interface in a web browser. The browser's address bar displays `localhost:8888/notebooks/vla-cont-tutorial.ipynb`. The notebook's title bar indicates it is titled `vla-cont-tutorial` and was last checkpointed 39 minutes ago. The interface includes a menu bar with `File`, `Edit`, `View`, `Insert`, `Cell`, `Kernel`, and `Help`. A toolbar below the menu bar contains icons for file operations, running, and other notebook functions. The main area contains a code cell labeled `In [27]:` with the following Python code:

```
bandpass(vis='3c391_ctm_mosaic_10s_spw0.ms', caltable='3c391_ctm_mosaic_10s_spw0.B0',
         field='J1331+3030', spw='', refant='ea21', combine='scan',
         solint='inf', bandtype='B',
         gaintable=['3c391_ctm_mosaic_10s_spw0.antpos',
                   '3c391_ctm_mosaic_10s_spw0.G0',
                   '3c391_ctm_mosaic_10s_spw0.K0'])
```

Below the code cell is a blue button labeled `Show log`. A red arrow points from the code cell to this button. The output of the code is a log of messages from the `bandpass` and `Calibrator` classes, showing the process of opening the measurement set, selecting channels, and applying calibration.

```
2017-10-12 10:24:17 INFO bandpass::calibrator::open Opening MS: 3c391_ctm_mosaic_10s_spw0.ms for calibration.
2017-10-12 10:24:17 INFO bandpass::Calibrator:: Initializing nominal selection to the whole MS.
2017-10-12 10:24:17 INFO bandpass::calibrator::reset Resetting solve/apply state
2017-10-12 10:24:17 INFO bandpass::calibrator::setdata Beginning selectvis--(MSSelection version)-----
2017-10-12 10:24:17 INFO bandpass::calibrator::reset Resetting solve/apply state
2017-10-12 10:24:17 INFO bandpass::Calibrator::selectvis Performing selection on MeasurementSet
2017-10-12 10:24:17 INFO bandpass::Calibrator::selectvis+ Selecting on field: 'J1331+3030'
2017-10-12 10:24:17 INFO bandpass::Calibrator::selectvis By selection 845379 rows are reduced to 31964
2017-10-12 10:24:17 INFO bandpass::Calibrator::selectvis Frequency selection: Selecting all channels in all spws.
2017-10-12 10:24:17 INFO bandpass::calibrator::setdata chanmode=none nchan=1 start=0 step=1 mStart='0km/s'
mStep='0km/s' msSelect=""
2017-10-12 10:24:17 INFO bandpass::calibrator::setapply Beginning setapply--(MSSelection version)-----
2017-10-12 10:24:17 INFO bandpass::Calibrator::setapply(type, applypar) Arranging to APPLY:
2017-10-12 10:24:17 INFO bandpass::: (KAntPos Jones: Overriding with spwmap=[0] since KAntPos Jones is not spw-
dependent)
2017-10-12 10:24:17 INFO bandpass::: (KAntPos Jones: Enforcing calWt()=False for phase/delay-like terms)
2017-10-12 10:24:17 WARN bandpass::: No VLATrDelCorr keyword in the antpos caltable; turning trop delay correction OFF.
2017-10-12 10:24:17 INFO bandpass::Calibrator::setapply(type, applypar) . KAntPos Jones:
table=3c391_ctm_mosaic_10s_spw0.antpos select= interp=linear spwmap=[0] calWt=false
2017-10-12 10:24:17 INFO bandpass::calibrator::setapply Beginning setapply--(MSSelection version)-----
2017-10-12 10:24:17 INFO bandpass::Calibrator::setapply(type, applypar) Arranging to APPLY:
2017-10-12 10:24:17 INFO bandpass::Calibrator::setapply(type, applypar) . G Jones:
table=3c391_ctm_mosaic_10s_spw0.G0 select= interp=linear spwmap=[-1] calWt=true
2017-10-12 10:24:17 INFO bandpass::calibrator::setapply Beginning setapply--(MSSelection version)-----
2017-10-12 10:24:17 INFO bandpass::Calibrator::setapply(type, applypar) Arranging to APPLY:
2017-10-12 10:24:17 INFO bandpass::: (K Jones: Enforcing calWt()=False for phase/delay-like terms)
2017-10-12 10:24:17 INFO bandpass::Calibrator::setapply(type, applypar) . K Jones: table=3c391_ctm_mosaic_10s_spw0.K0
select= interp=linear spwmap=[-1] calWt=false
2017-10-12 10:24:17 INFO bandpass::calibrator::setsolve Beginning setsolve--(MSSelection version)-----
```



vla-cont-demo

+

← → ↻ 🏠

🔍 jupyter.jive.nl/user/f4wycZFsiIeb/notebooks/vla-cont-demo.ipynb


⋮ 📄 ⌵

jupyter vla-cont-demo (autosaved)


File Edit View Insert Cell Kernel Help

Not Trusted Casa

📁 + 🔍 📄 ⬆ ⬇ ⬆ ⬇ ▶ Run ⬛ ↺ ▶ Code



The notebook interface to CASA was developed by Aard Keimpema at the [Joint Institute for VLBI ERIC \(JIVE\)](#). Installation in the sandbox interface (based on [timpnb](#)) was done by Tammo Jan Dijkema at [ASTRON](#). This work is a part of the [ASTERICS](#) project, Astronomy ESFRI & Research Infrastructure Cluster ASTERICS - 653477.



This notebook interface was built on top of CASA 5.3.0. Adaptions were made to run it on a newer version of Python than the one distributed with CASA. Future development of this notebook, or upgrading the underlying CASA version, is currently being planned.

In [1]:

`casa['build']['version']`

Out[1]:

`'5.3.0-143 '`

http://jupyter.jive.nl

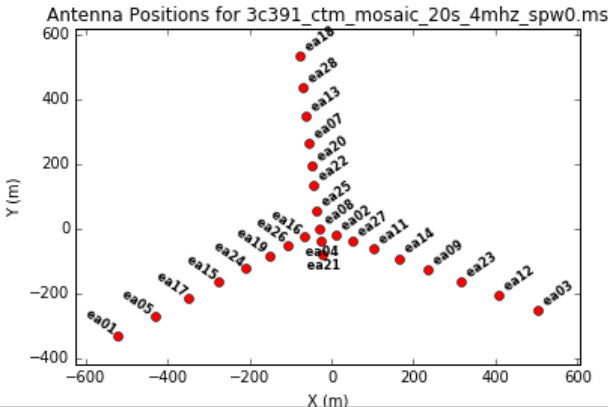
To get a sense of the array, as well as identify an antenna for later use in calibration, we use the task `plotants`.

In [3]:

`plotants(vis='3c391_ctm_mosaic_20s_4mhz_spw0.ms',  
figfile='plotants_3c391_antenna_layout.png')  
clearstat() # This removes the table lock generated by plotants in script mode`

Number of points being plotted: 26

Antenna Positions for 3c391\_ctm\_mosaic\_20s\_4mhz\_spw0.ms

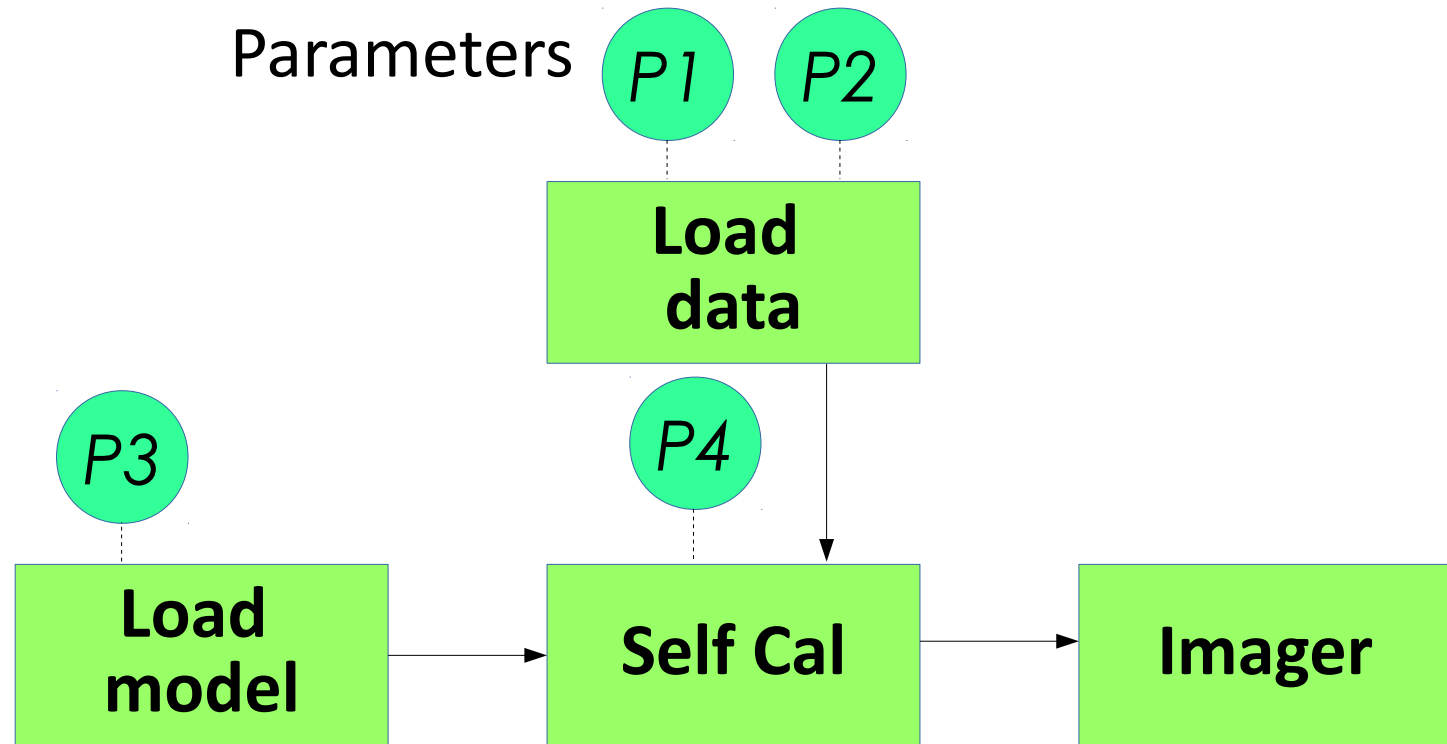


# CASA for Jupyter

- NRAO CASA distribution is entirely self-contained, it is essentially a linux distribution
- **Too old for Jupyter**, many conflicting packages, e.g. Matplotlib, IPython, ....
- We created a custom build of CASA using based on latest Python
- Distributed as **Docker** and **Singularity** containers.
  - *docker pull penngwyn/jupytercasa*
  - *singularity pull shub://aardk/jupyter-casa*

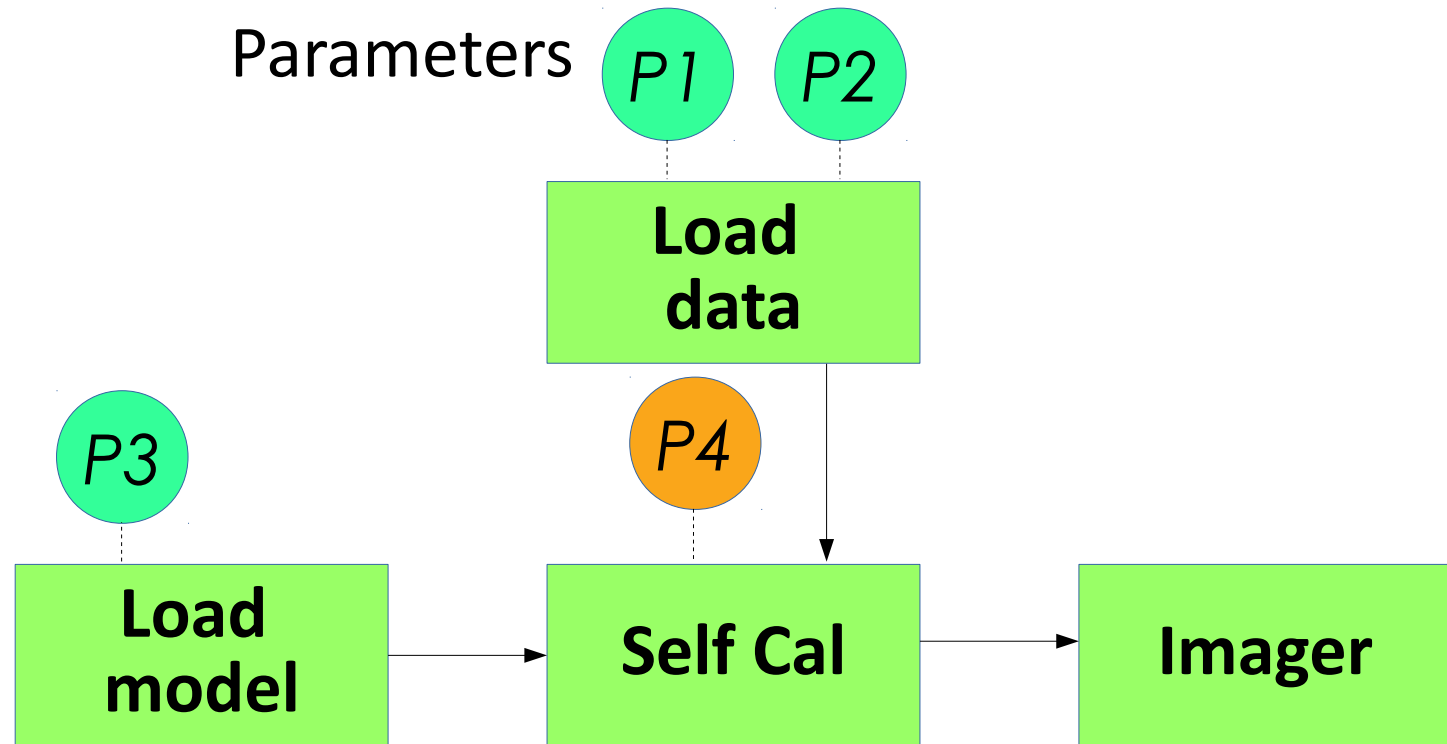


# Minimal Re-computation framework



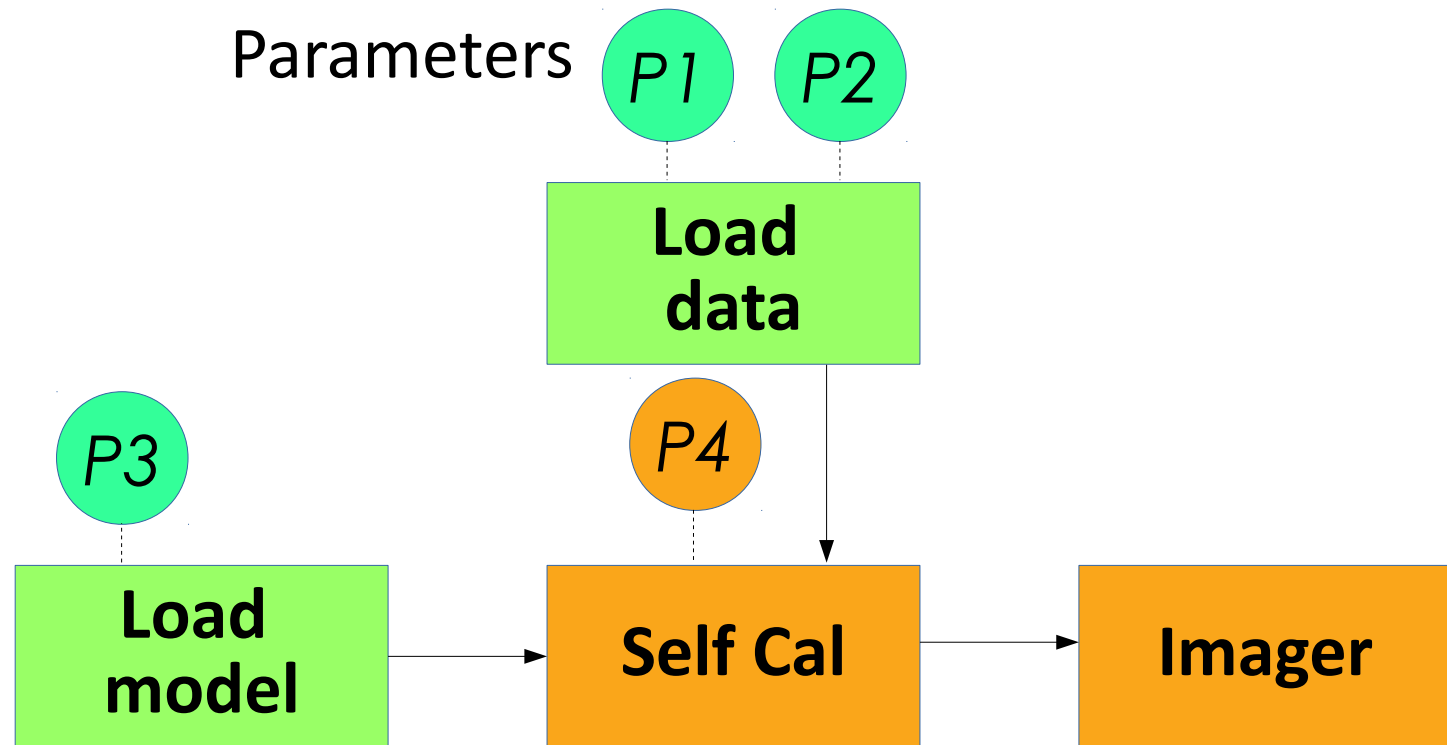
- Astronomy and Computing, **25**, 133, 2018 (arXiv:1711.06124 )
  - U. Cambridge: Bojang Nikolic, JIVE: Des Small, and Mark Kettenis
- Generate dependency graphs
- Cache results efficiently using ZFS copy-on-write
- Implemented on separate *min-recomputation* branch

# Minimal Re-computation framework



- Astronomy and Computing, **25**, 133, 2018 (arXiv:1711.06124 )
  - U. Cambridge: Bojang Nikolic, JIVE: Des Small, and Mark Kettenis
- Generate dependency graphs
- Cache results efficiently using ZFS copy-on-write
- Implemented on separate *min-recomputation* branch

# Minimal Re-computation framework



- Astronomy and Computing, **25**, 133, 2018 (arXiv:1711.06124 )
  - U. Cambridge: Bojang Nikolic, JIVE: Des Small, and Mark Kettenis
- Generate dependency graphs
- Cache results efficiently using ZFS copy-on-write
- Implemented on separate *min-recomputation* branch

# Conclusions

- We have implemented Jupyter kernel for CASA suitable for pipelines
- We provide both Docker and Singularity images for easy deployment
- Minimal re-computation framework enables efficient reliable iterative pipelines.
- *<https://github.com/aardk/jupyter-casa>*
- *<http://jupyter.jive.nl/>*

# Acknowledgement

- H2020-Astronomy ESFRI and Research Infrastructure Cluster (Grant Agreement number: 653477).